John von Neumann Institute for Computing

Services Integration and Task-scheduling in Bioinformatics Grids

S. Ramírez, E. de-Andrés, I. Navas-Delgado, A.J. Pérez, J. Aldana, O. Trelles

published in

Parallel Computing:
Current & Future Issues of High-End Computing,
Proceedings of the International Conference ParCo 2005,
G.R. Joubert, W.E. Nagel, F.J. Peters, O. Plata, P. Tirado, E. Zapata (Editors),
John von Neumann Institute for Computing, Jülich,
NIC Series, Vol. **33**, ISBN 3-00-017352-8, pp. 735-742, 2006.

http://www.fz-juelich.de/nic-series/volume33

Services Integration and Task-scheduling in Bioinformatics Grids

Sergio Ramírez[a], Enrique de-Andrés[b], Ismael Navas-Delgado[c], Antonio J. Pérez[a], José Aldana[c], Oswaldo Trelles[a]

[a]Architecture of Computer Department, University of Malaga, Spain

[b]Scientist Park of Madrid, Madrid, Spain

[c]Computacional Languages Department, University of Malaga, Spain

Abstract

We report the design and structured development of a computational, data and services Grid environment at the National Institute for Bioinformatics (INB) in Spain. The design allows the easy integration of Web-based services particularly suitable for collaborative work in which users share tools, data and services. The system offers a view of public and proprietary databases as a single data source where services are readily available for enhancing data processing. The availability of a broad collection of services together with several instances of the same service on different sites allows parallel computing to be drawn over the Grid to take advantage of the huge computational resources available at the different nodes that conforms the INB. The computational load is distributed among the servers and results are coherently collected. A robust Grid-management module supplies queuing mechanisms; resources allocation, load monitoring, service-quality-based scheduling and implements fault tolerance procedures. By combining task-grained scheduling and buffering strategies to reduce idle time for load reposition and to take advantage of I/O overlapping, we have seen important increase of efficiency in the use of resources. Several tests have been performed using applications from the bioinformatics domain for which the adaptive scheduling strategy has shown its ability to produce a noticeable reduction on execution time.

Keywords: Data-service, data-Grid, bioinformatics, integration, task-scheduling, workflows

Prototype Availability: The INB prototype is available at http://www.inab.org/MOWServ/ Supplementary data on the developed tests is available at the help system.

1. Introduction

Currently, Grid computing [1] most probably represents the new generation of web-based technology. The use of such technology has promoted the development of distributed frameworks for high performance computing by means of intelligent integration of disperse and heterogeneous computational resources throughout high speed networks.

This new computing capability becomes of special interest in the bioinformatics domain where: (a) Highly diverse and impressive volume data collections are geographically disperse and have heterogeneous formats, making this plethora of interrelated information difficult to use; (b) data and services are frequently replicated in several sites; and (c) the park of installed hardware in traditional bio-wet-labs consist mainly of collections of commodity PCs. Under this scenery, Grid technology arises as a natural alternative to provide computational power to the bioinformatic community.

Several proposals have addressed the integration of bioinformatics resources. In fact, very popular sites like NCBI, EBI, ExPASy, etc.; represent a colossal effort to provide ”linked” access to a high number of interrelated data sources. However, since each service provider maintains its own web-interface style, parameter specifications and since is none standarized description about what the

input/outputs from a service are, it is difficult built up automatic service discovering and processing for the massive exploitation of Grid computational power.

Less known in the application domain but also addressing integration issues we can describe TAMBIS [15]. It makes use of an ontology that allows it to provide homogeneous layers that envelop data-bases to manage heterogeneity between sources and to provide a query interface to create and refine queries. The key idea of BioDataServer [16] for molecular database integration is a mediation architecture in which a wrapper exports some information about its source schema, data and query processing capabilities for each data source.

To meet the challenges of integrating and analysing diverse scientific data from the variety of domains within life sciences, IBM has developed a versatile platform solution, IBM DiscoveryLink [17]. With single query data access, the IBM DiscoveryLink software allows researchers to work with distributed data sources and diverse data formats. PISE [18] is a Web interface generator for molecular biology command-line driven programs, including: phylogeny, gene prediction, alignment, RNA, DNA and protein analysis, motif discovery, structure analysis and database searching programs. Its aim is to provide users with the equivalent of a basic Unix environment, with program combination, customisation and basic scripting through macro registration. EMBOSS [19] (”The European Molecular Biology Open Software Suite”) is a software analysis package specially developed for the needs of the molecular biology (e.g. EMBnet) user community. The software automatically copes with data in a variety of formats and even allows transparent retrieval of sequence data from the web.

More focused in the application domain, BioMOBY [2] is a project that proposes an architecture for the discovery and distribution of biological data, using web services. In this architecture data and services are decentralized. However, the resources are registered in a central location called MOBY central. BioMOBY objects are lightweight XML, and make up both the query and the response of a simple object access protocol (SOAP) transaction. The primary components of this architecture are MOBY Services (bioinformatics tools), MOBY Objects (input and output data in the services), MOBY Central (registry of all resources), and Object and Service hierarchies. This proposal includes the possibility of easily developing web services for publishing biological data. It introduces the use of web services for publishing and using biological data, but it is not exactly integration architecture. Taverna and Talisman [12] are in fact clients for running BioMOBY services. Both make extensive use of XML files in the Scufl representation language. Finally, myGrid [3] is an ongoing project to develop an integrated platform for bioinformatics services, with a good definition of biological objects.

On the management of Grid resources and more focused on the scheduling problem in Grid environments Nimrod-G [4] allows the distribution of multiple runs of the same process with different parameters, under static Grid configuration. This problem is solved by GRaDS scheduler [5] [6] by dynamically adapting the strategy when new resources are available. Finally, Condor-G modifies its standard scheduler to incorporate new machines connected by Globus. None of these schedulers incorporates fault tolerance concerns, neither dynamic load balancing as a function of the real capacity of each component of the Grid, these concerns are both central aspects of this work. Recently our group develop a Globus-Condor-based Grid environment in which an initial proposal were addressed [7]. Now we report a robust and stable implementation of these ideas at the INB-Spain.

We have been working from the point of view of implementing a versatile client with single and uniform user interfaces to access and present information from multiple online services and databases; and working in an internal infrastructure to integrate different and geographically distributed Web Services providing a way for fast and intuitive ”wiring” of services to create virtual

complex, distributed and powerful bioinformatics machines. Since the platform has been endowed with the ability to manage huge computational resources, dynamic scheduling and fault tolerance techniques have been used to allow massive exploitation of resources.

2. The INB web-services platform

The National Institute for Bioinformatics (INB) in Spain have addressed the integration problem throughout the design of a simple, dynamic and extensible platform to represent, recover, process, integrate and discovering knowledge. To integrate geographically distribute resources a Grid-enable system has been built on top of BioMOBY API, offering a view of the system databases as a single data source where services are readily available for enhancing data processing. Description of input/output objects is coordinated and standardised by means of an object-ontology in such a way that services can communicate among them, wiring natural bioinformatics workflows. Automatic interfaces and help system builders have been incorporated into the architecture to uniform and facilitate user communication. Beyond traditional bioinformatics platforms, data persistence system, user management and scheduling abilities have drawn a new generation of bioinformatics platforms.

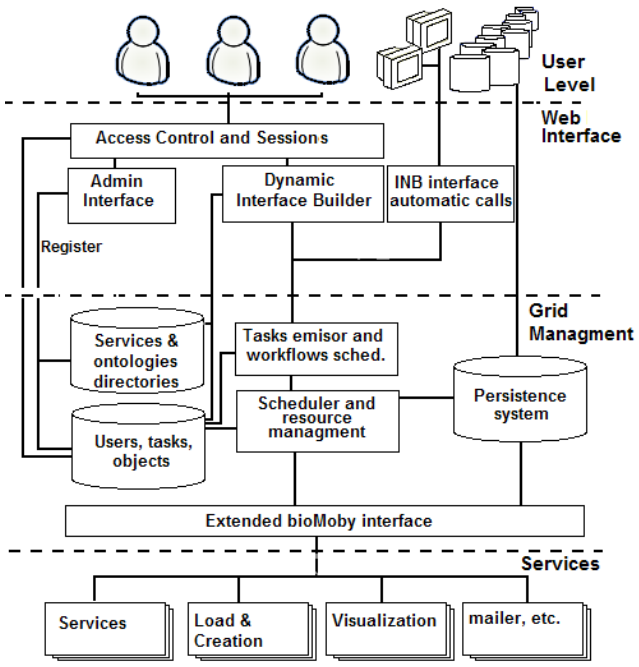

Figure 1: INB Architecture. On top, external interface including authentication and proprietary data control. Scheduling, with analysis of task-dependencies for workflows; data persistence and the capacity to use collections is offered in the Grid management level. Finally an extended BioMoby interface; including asynchronous services calls and errors management

In short the INB project aims:

- To complement and extend the BioMOBY standard to provide access to information resources (data, service and computational power), independently of a user's knowledge of their original existence.
- To combine retrieved information from these resources based on common ontologies, wiring resources and tools that in other way could not easily be linked. The workflows approach -linking several services to solve specific biological problems- opens new perspectives to the biological information analysis in Bioinformatics.
- To make interfaces simple to use, self contained and intuitive, eliminating the need for a high level of tacit knowledge. This includes displaying processed information in a consistent faceted way.
- To facilitate the incorporation of high quality, highly demanded new services we provide an internal mechanism for the efficient use of computational resources. Scalability in the platform is provided by an intelligent computational load distribution mechanism.

The scheme in 1 depicts the INB system architecture organised in three main levels, with the user minimally armed with a Web-browser, and demanding services to process their collection of biological data: (a) a web-interface on top of the architecture facilitates communication between the user and the platform (b) the architecture core including services' interface though BioMOBY API; and (c) on bottom of the scheme the services' providers.

A web interface manages user sessions with authentication mechanism. An automatic web interface builder is able to dynamically built-on interfaces for browsing data, services and namespaces (mostly associated with data containers). The catalogue offer is deployed in the form of a browseable tree from which the user gains access to procedures. In the same way automatic interfaces are built for services parameters and a generic creation service allows new objects be incorporated into the system. Up- and download procedures have also been incorporated in the platform. Noteworthy to observe, the system produce an auto-generated service help that includes training examples for getting started with each service (made available during the service registration procedure).

3. Grid Management at the INB platform

At internal level, once a service has been launched, the system provides notification about the progress status of services, -including historical record of executed tasks-, together with the relationships between input data, applied service and output data. Error notification has been incorporated in the system by extending the BioMOBY protocol. Frequently output data become the input for new services. The GUI provides a specific list of suitable services that can be applied.

Different services and multiple instances of the same service can be installed at the same or in different sites, offering computational power at different scales (ranging from simple PC servers to high cost multiprocessors platforms). Since a high number of users are expected (optimistically we look at some notable sites with more than million access per year), a pool of tasks to be solved at each moment will be the natural working scenery. Under this situation, scheduling arise as a natural need.

The Task-scheduler works over this pool of tasks and uses the map of services to choose the best server to solve the task. A task-dispatcher is in charge of launching a worker-process to communicate with the service to solve the task. Noteworthy to observe the fact that new servers and services becomes available as soon as the registry procedure is complete. Since servers can conform a sub-Grid, the scheduler transfers the job to its front-end and maintains the record of the pending jobs and the machines in charge of it. A buffering technique has been implemented, to send work in advance, avoiding delay for job reposition from the scheduler and benefit from I/O overlapping in the same machine with replicate services, which is in fact important due to bioinformatics applications are typically I/O bounded.

Load distribution is performed in a dynamic and adaptive fashion, as fast as new tasks are available. The current configuration is used to know the computational power available at a given time. The system evaluates the CPU cost of each task and adjusts predictions when the tasks are reported from the services. Load size is computed as a function of the tasks CPU cost and the quality of the service, estimated as the historic response time.

With some more detail, to perform the load distribution the following considerations are contemplated:

1. The architecture is composed of several nodes, which at the same time can form an internal sub-Grid.

2. The model is able to split a complex task in their simple components (workflows and collection of tasks).
3. Tasks are queued in a pool from which the scheduler has access to them. Tasks arrive to the queue from a tasks dependencies analyser
4. The scheduler maintains a dynamic control of the Grid configuration.
5. Fault tolerance mechanisms allow the detection of services which do not respond to the assigned task.
6. Dynamic task distribution is performed based on the quality level of the service (response time).
7. Idle time for load reposition is avoided by sending new tasks in advance (buffering).

4. Task scheduling

Work requests are analysed by a workflows and complex-task module producing individual tasks and control data dependencies among them. The scheduler works over this pool and uses a map to choose the best server to solve the task. The configuration-module dynamically traces nodes, incorporating new resources and works in close collaboration with the fault tolerance module who is in charge of the pending tasks, and it is able to re-insert tasks on the pool. Since servers can conform a sub-Grid the scheduler transfers the job to its front-end. A buffering technique [8] [9] has been implemented, to send work in advance, avoiding delay for job reposition from the scheduler. Load distribution is performed in dynamic and adaptive fashion, as fast as new tasks are available. The current configuration is used to know the computational power available at a given time. The system evaluates the CPU cost of each task and adjusts predictions when the tasks are reported from the services. Load size is computed as a function of the tasks CPU cost and the quality of the service, estimated as the historic response time (a configurable parameter).

The load distribution is estimated using the average speed observed in each service (v_i) relative to the global average speed in the system (v_{global}). The scheduler determines the size tasks (Q) to be sent to the server based on the initial task size (Q_{inic}) $Q = \frac{v_i}{v_{global}} Q_{inic}$. This distribution scheme increases the load to those servers with response-time faster than the global speed and reduces the load in the converse situation. To compute the average speed of execution the recent history is given more weight than the old one. This average speed (v_i) is computed as: $v_i = v_i' d + v_{exec}(1-d)$; where d (decay) is a configurable value (0.8 by default) related to how fast the older values are forgotten by the system, and thus it controls the velocity of adaptation of the scheduler to the changes in the quality of the service.

5. Results

The current INB interface deploys a set of services, which make possible to carry out different analysis on biological data. These different tools can be automatically applied over a set of data to produce a complete analysis to solve complex biological problems using the same platform. In this line we present a practical example to solve a phylogenetic study using an amino acid sequence as the starting point (see 2). Homology search is conducted (Blast services) to obtain similar sequences with a common evolutionary history. Output from this service contains a set of putative homologous sequences to the query. A new service is linked (Clustalw from Blast) to build-up a multiple alignment with the most similar reported sequences. Finally a phylogenetic tree is obtained using CreateTreeFromClustalw service highlighting the relations between all the sequences. Further, changing the service parameters for deeper study can customise this generic workflow.

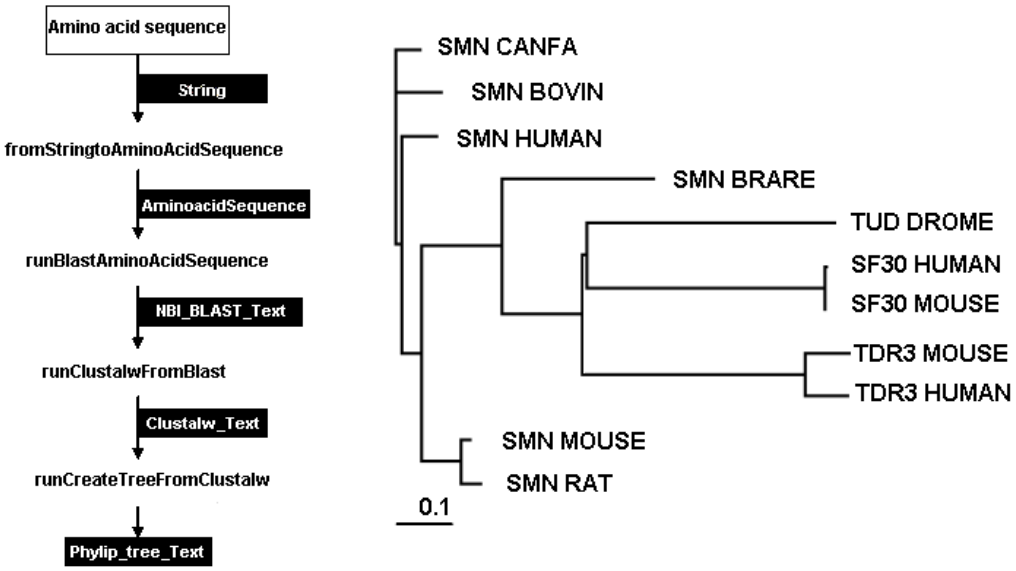

Figure 2. Homology search and phylogenetic study workflow. On the left, the grey box shows the input data; ellipses correspond to INB services, and black boxes show the input and output objects of each service in this workflow. Final and partial results remain available at the data persistence system. On the right the resulting phylogenetic tree showing the relationships among the protein sequence related to SMN_HUMAN.

This workflow has been tested with the human survival motor neuron protein (SMN; Accession Number: Q16637) running against SWISS-PROT database and a relaxed e-value was used as threshold to select distantly related homologous hits to carry out the multiple alignment. Interesting results are reported such as the fact that several related sequences show a common domain called 'Tudor domain', first identified as fragment repeats in Drosophila melanogaster [13] [14]. Surprisingly, as result of our analysis this large Drosophila protein (TUD_DROME) appears separated from the remaining proteins in the phylogenetic tree (2) suggesting a relationship with splicing factors (SF30 proteins). In short, these results can conclude that the Drosophila protein, which is required during ogenesis for the formation of primordial germ cells and for normal abdominal segmentation, could be a splicing factor assisting this process.

In the second line of work we present a parallel solution for massive Blast searches using the Grid resources at the INB. Initial tests have been performed using the computational resources available at the Scientific Park of Madrid (PCM). The PCM deploys a super cluster SGI Altix 3700 that is based on a shared memory multiprocessor architecture with 24 processors Intel Itanium2 at 1600MHz and 6Mb cache each one. The memory and disk storage are 40Gb and 1Tb respectively. The PCM also has three Dual Intel Xeon Servers at 2.8 GHz with 1 GB of RAM memory.

In this test we have used the sequence of the bacterium *Buchnera aphidicola str. Bp* (Baizongia pistaciae) with Accession Number NC_004545, whose proteome has 504 proteins and it is considered one of the known organisms with the minimal gene number needs for symbiotic life. In this way this proteome constitutes a diverse set of proteins representing roughly the different amino acid sequences in the nature.

Two experiments have been conducted to study the (a) effect of using a parallel Blast implementation to solve each task; (b) the influence of the task and problem size. In both cases we have used advanced delivery of jobs (to avoid idle time) and the fault-tolerance module was active (to complete

the set of tasks and make the experiments comparable). The main results can be drawn as follow: (1) execution time is significantly reduced even for single sequences tasks, until a number of processors are enough (6 for a medium size sequence of 400 aa). However, the combined use of parallel Blast and parallel tasks allows a high number of PEs to be used with promising results in reduction time.

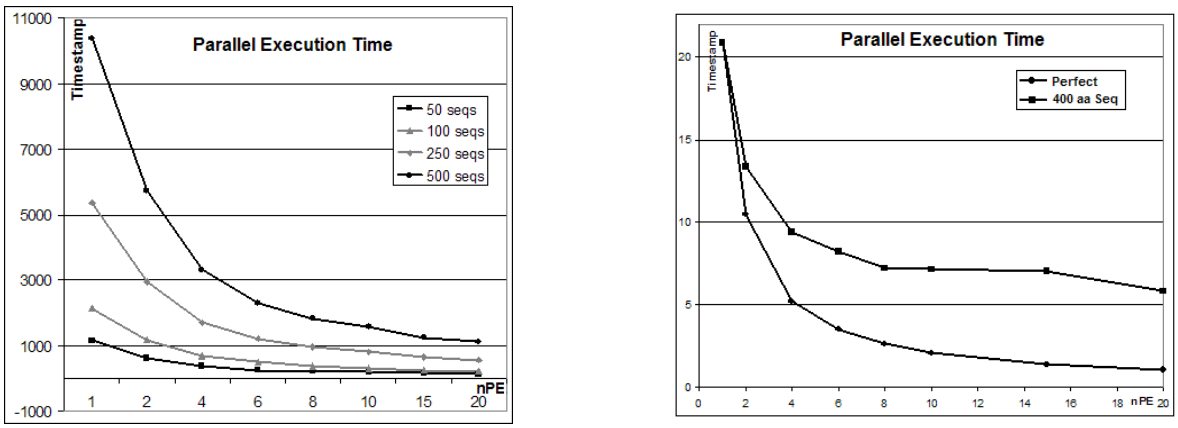

Figure 3. Parallel execution time, using parallel Blast implementation (on the left) and combining parallel Blast and parallel tasks in different number of processors (on the right).

As expected, for low load (50 and 100 sequences datasets) the benefits of using multiprocessors are limited due to the overhead produced by communication time. Results compares quite well against those obtained using single-processors with high efficiencies.

6. Conclusions

In summary we have developed a dynamic and flexible integration architecture that provide fast and intuitive ”wiring” of services thus creating virtual complex, distributed and powerful bioinformatics machines. Based on semantic interconnection concepts the platform integrates into workflows diverse traditional databases and various processing mechanisms developed by different users and groups through a web-based interface expanding the functionality of current services and enabling the easy incorporation of new procedures to customize the system for specific concerns. Additionally by supplying computational power the INB architecture makes easier the challenge of discovering new knowledge from the avalanche of new biological data.

An adaptive scheduling for task-grained execution in Grid environments has been presented. The scheduling policy has been applied to the resolution of a common bioinformatics problem and each scheduling module has been individually analysed for their influence on efficiency improvement: task buffer size, task size; initial load size; fault-tolerance, etc. These results compare quite well with those obtained using static scheduling with very encouraging results.

The usefulness of the INB-architecture is demonstrated both at the level of integrating diverse services to produce a complete view of a given biological process and to obtain results with a significant reduction of elapsed time compared with equivalent installations.

7. Acknowledgements

This work has been partially supported by grant ”GNV5-Bioinformática Integrada” from Genoma-España.

References

[1] I. Foster, C. Kesselman, and S. Tuecke: The anatomy of the grid: Enabling scalable virtual organizations. Supercomputer Applications, 2001 (http://www.globus.org/research/papers/anatomy.pdf).

[2] Wilkinson, MD, Links, M.: ”BioMOBY: an open-source biological web services proposal”.Briefings In Bioinformatics 3:4. 331-341.2002 (www.biomoby.org)

[3] R. Stevens, A. Robinson y C. A. Goble: ”myGrid: Personalised Bioinformatics on the Information Grid”. In proceedings of 11th International Conference on Intelligent Systems in Molecular Biology. Brisbane, Australia. Bioinformatics vol. 19 Sup. 1 2003.

[4] Abramson, D.: ”High performance parametric modelling with Nimrod/G: Killer application for the global Grid?”. In proceedings of the Seventh International Symposium on High Performance Distributed Processing Symposium (IPDPS’00).2003

[5] Berman, F.: ”The GrADS Project: Software support for high-level Grid application development”. International Journal of Supercomputer Applications 15, 4, 327-344. 2001

[6] H. Dail: ”A Decoupled Scheduling Approach for the GrADS Environment”. Proceedings of SC 2002, Baltimore.2002.

[7] Manuel Hidalgo Conde, Andr’es Rodr’iguez, Sergio Ram’irez and Oswaldo Trelles: ”Adaptive Task Scheduling in Computational GRID environments”. European GRID Conference; Amsterdan, Holland.2005

[8] Trelles-Salazar, O.; Zapata. E. and Carazo, J.M.: ”Mapping Strategies for Sequential Sequence Comparison Algorithms on LAN-based Message Passing Architectures”. High Performace Computing and Networking (HPCN-Europe’94), Munich-Alemania.1994

[9] Trelles-Salazar, O. Zapata, E.L. and Carazo J.M.: ”On an efficient parallelization of exhaustive sequence comparison algorithms”. Computer Applications in BioSciences 10(5):509-511.1994

[10] Rodriguez, A. Prez-Pulido, A:, Thode,G.; Carazo,JM. y Trelles,O.: ”Mining Low-level similarity signals from sequence databases”.High Performance Computers Applied to BioInformatics and Computational Biology. 4th Conference on Systematics, cybernetics and Informatics, SCI’2000; Orlando, Florida, USA.2000

[11] Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ.: ”Gapped BLAST and PSI-BLAST: a new”.1997

[12] Oinn, Tom; Matthew Addis, Justin Ferris, Darren Marvin, Martin Senger, Mark Greenwood, Tim Carver, Kevin Glover, Matthew R. Pocock, Anil Wipat and Peter Li: ”Taverna: A tool for the composition and enactment of bioinformatics workflows Bioinformatics Journali”.20(17) pp 3045-3054.2004

[13] Ponting C.P.: ”Tudor domains in proteins that interact with RNA. Trends Biochem”. Sci. 22:51-52. 1997

[14] Callebaut I., Mornon J.P.: ”The human EBNA-2 coactivator p100: multidomain organization and relationship to the staphylococcal nuclease fold and to the tudor protein involved in Drosophila melanogaster development”. Biochem. J. 321:125-132. 1997

[15] Stevens et alt.: ”TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources”. Bioinformatics, 16:2 PP. 184-186.

[16] Lange et alt.: ”A computational Support for Access to Integrated Molecular Biology Data”.

[17] DeCarlo, J.: ”IBM Life Sciences Solutions: Turning Data into Discovery with DiscoveryLink”. ISBN 0738423254. 2002

[18] Letondal, C.: ”A Web interface generator for molecular biology programs in Unix. Bioinformatics”, Oxford University Press, 17(1), pp 73-82. 2001

[19] Rice,P. et al.: ”EMBOSS: the european molecular biology open software suite”. Trends Genet., 16, 276-277. 2000